Efficient SDP Inference for Fully-connected CRFs Based on Low-rank Decomposition

Peng Wang, Chunhua Shen, Anton van den Hengel

Abstract

Conditional Random Fields (CRF) have been widely used in a variety of computer vision tasks. Conventional CRFs typically define edges on neighboring image pixels, resulting in a sparse graph such that efficient inference can be performed. However, these CRFs fail to model long-range contextual relationships. Fully-connected CRFs have thus been proposed. While there are efficient approximate inference methods for such CRFs, usually they are sensitive to initialization and make strong assumptions. In this work, we develop an efficient, yet general algorithm for inference on fully-connected CRFs. The algorithm is based on a scalable SDP algorithm and the low-rank approximation of the similarity/kernel matrix. The core of the proposed algorithm is a tailored quasi-Newton method that takes advantage of the low-rank matrix approximation when solving the specialized SDP dual problem. Experiments demonstrate that our method can be applied on fully-connected CRFs that cannot be solved previously, such as pixel-level image co-segmentation.

I. Introduction

Semantic image segmentation or pixel labeling is a key problem in computer vision. Given an image, the task is to label every pixel against one or multiple pre-defined object categories. It is clear that to achieve satisfactory results, one must exploit contextual information. Scalability and speed of the algorithm are also of concerns, if we are to design an algorithm applicable to high-resolution images.

Conditional random fields (CRFs) have been one of the most successful approaches to semantic pixel labeling, which solves the problem as maximum a posteriori (MAP) estimation. Standard CRFs contain unary potentials that are typically defined on low-level features of local texture, color, and locations. Edge potentials, which are typically defined on 4- or 8-neighboring pixels, consist of smoothness terms that penalize label disagreement between similar pixels, and terms that model contextual relationships between different classes. Although these CRF models have achieved encouraging results for segmentation, they fail to capture long-range contextual information.

In the literature, fully-connected CRFs have been proposed for this purpose. The main challenge for inference on fully-connected CRFs stems from the computational cost. A fully-connected CRF over N image pixels has N^2 edges. Even for a small images with a few thousand pixels, the number of edges can be a few million. Although there have been a variety of methods for MAP estimation [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], they are usually computationally infeasible for such cases. The authors of [11], [12] have proffered an efficient mean field approximation method for MAP inference in multi-label CRF models with fully connected pairwise terms. A filter-based method is used to accelerate the computation. The assumption is that the pairwise terms are in the form of a weighted mixture of Gaussian kernels such that fast bilateral filtering can be applied. For a special type of fully-connected CRF, in which the edge potentials are defined to capture the spatial relationships among different objects, and only depend on their relative positions (that is they are spatially stationary), an efficient inference algorithm was developed in [13]. The method proposed in [14] can be applied on generalized RBF kernels, instead of the original Gaussian kernels. Note that there is still a strong assumption in [14], which limits the practical value of this method.

In general, semidefinite programming (SDP) relaxation provides accurate solutions for MAP estimation problems, but it is ususally computationally inefficient (see [3] for the comparison of different relaxation methods). Standard interior-point methods require $\mathcal{O}(m^3+mn^3+m^2n^2)$ flops to solve a generic SDP problem in worst-case, where n and m are the semidefinite matrix dimension and the number of constraints respectively. Recently, several scalable SDP

The authors are with School of Computer Science, University of Adelaide, Australia; and ARC Centre of Excellence for Robotic Vision. Correspondence should be addressed to C. Shen (chunhua.shen@adelaide.edu.au).
A conference version of this work is appearing in Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2015.

$\mathbf{X}$	A matrix (bold upper-case letters).
$\mathbf{x}$	A column vector (bold lower-case letters).
$\mathcal{S}^n$	The space of $n \times n$ symmetric matrices.
$\mathcal{S}^n_+$	The cone of $n \times n$ symmetric positive semidefinite (SPSD) matrices.
$\mathbb{R}^n$	The space of real-valued $n \times 1$ vectors.
$\mathbb{R}^n_+$,$\mathbb{R}^n_-$	The non-negative and non-positive orthants of $\mathbb{R}^n$.
$\mathbf{I}_n$	The $n \times n$ identity matrix.
$\mathbf{0}$	An all-zero vector with proper dimension.
$\mathbf{1}$	An all-one vector with proper dimension.
$\leq, \geq$	Inequality between scalars or element-wise inequality between column vectors.
$\mathrm{diag}(\mathbf{X})$	The vector of the diagonal elements of the input matrix $\mathbf{X}$.
$\mathrm{Diag}(\mathbf{x})$	The $n \times n$ diagonal matrix whose main diagonal vector is the input vector $\mathbf{x}$.
$\mathrm{trace}(\cdot)$	The trace of a matrix.
$\mathrm{rank}(\cdot)$	The rank of a matrix.
$\delta(cond)$	The indicator function which returns 1 if $cond$ is ture and 0 otherwise.
$\lVert\cdot\rVert_F$	Frobenius-norm of a matrix.
$\langle \cdot, \cdot \rangle$	Inner product of two matrices.
$\circ$	Hadamard product of two matrices.
$\otimes$	Kronecker product of two matrices.
$\nabla \mathrm{f}(\cdot)$	The first-order derivative of function $\mathrm{f}(\cdot)$.
$\nabla^2 \mathrm{f}(\cdot)$	The second-order derivative of function $\mathrm{f}(\cdot)$.
$n!$	The factorial of a non-negative integer n.

TABLE I: Notation.

methods have been proposed for MAP estimation. Huang *et al*. [15] proposed an alternating direction methods of multipliers method (ADMM) to solve large-scale MAP estimation problems. Wang *et al*. [16] presented an efficient dual approach (refer to as SDCut), which can also be applied for MAP estimation. However, their methods still cannot be applied directly to large-scale fully-connected CRFs.

There are two key contributions in this work:
(i) An efficient low-rank SDP approach (based on SDCut) for MAP estimation is proposed for MAP estimation in large-scale fully-connected CRFs. Several significant improvements over SDCut are presented, which makes SDCut much more scalable. The proposed SDP method also overcomes a number of limitations of mean field approximation, which provide more stable and accurate solutions.
(ii) Low-rank approximation methods for SPSD kernels (whose kernel matrix is symmetric positive semidefinite) is seamlessly integrated into the proposed SDP method, and used to accelerate the most computational expensive part of the proposed SDP method. The use of low-rank approximation relaxes the limitation on the pairwise term from being (a mixture of) Gaussian kernels to all symmetric positive-semidefinite kernels. The low-rank approximation method can be also used to replace the filter-based method in [11] for mean field approximation.

Thus our method is much more general and scalable, which has a much broader range of applications. The proposed SDP approach can handle fully-connected CRFs of #states $\times$ #variables up to 10^6. In particular, we show that on an image co-segmentation application, the fast method of [11] is not applicable while our method achieves superior segmentation accuracy. To our knowledge, our method is the first pixel-level co-segmentation method. All previous co-segmentation methods have relied on super-pixel pre-processing in order to make the computation tractable. Wang *et al*. [17] and Frostig *et al*. [18] also proposed efficient approaches which find near-optimal solutions to SDP relaxation to MAP problems. The main difference is that their methods solve (generally *nonconvex*) quadratically constrained quadratic programs by projected gradient descent, while ours uses quasi-Newton methods to solve a *convex* semidefinite least-square problem. Notation is listed in Table I.

II. PRELIMINARIES

A. Fully-connected Pairwise CRFs with SPSD kernels

Consider a random field over N random variables $\mathbf{x} = [x_1, x_2, \ldots, x_N]^\top$ conditioned on the observation $\mathbf{I}$. Each variable can be assigned a label from the set $\mathcal{L} := \{1, \ldots, L\}$. The energy function of a CRF $(\mathbf{I}, \mathbf{x})$ can be expressed by the following Gibbs distribution:

$$\mathrm{P}(\mathbf{x}|\mathbf{I}) := \frac{1}{Z(\mathbf{I})} \exp(-\mathrm{E}(\mathbf{x}|\mathbf{I})), \tag{1}$$

where $\mathrm{E}(\mathbf{x}|\mathbf{I})$ denotes the Gibbs energy function w.r.t. a labelling $\mathbf{x} \in \mathcal{L}^N$, and $Z(\mathbf{I}) := \sum_{\mathbf{x}\in\mathcal{L}^N} \exp(-\mathrm{E}(\mathbf{x}|\mathbf{I}))$ is the partition function. In the rest of the paper, the conditioning w.r.t. $\mathbf{I}$ is dropped for notational simplicity.

Assuming $\mathrm{E}(\mathbf{x})$ only contains unary and pairwise terms, the MAP inference problem for the CRF $(\mathbf{I}, \mathbf{x})$ is equivalent to the following energy minimization problem:

$$\min_{\mathbf{x}\in\mathcal{L}^N} \mathrm{E}(\mathbf{x}) := \sum_{i\in\mathcal{N}} \psi_i(x_i) + \sum_{i,j\in\mathcal{N}, i<j} \psi_{i,j}(x_i, x_j), \tag{2}$$

where $\mathcal{N} := \{1, \ldots, N\}$, and $\psi_i : \mathcal{L} \to \mathbb{R}, \forall i \in \mathcal{N}$ and $\psi_{i,j} : \mathcal{L}^2 \to \mathbb{R}, \forall i,j \in \mathcal{N}, i \neq j$ correspond to the unary and pairwise potentials respectively.

The pairwise potentials considered in this paper can be written as:

$$\psi_{i,j}(x_i,x_j) := \mu(x_i,x_j) \sum_{m=1}^{M} w^{(m)} \mathrm{k}^{(m)}(\mathbf{f}_i, \mathbf{f}_j), \tag{3}$$

where $\mathbf{f}_i, \mathbf{f}_j \in \mathbb{R}^D$ indicate D-dimentional feature vectors corresponding to variables x_i and x_j respectively. $\mathrm{k}^{(m)} : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ denotes the m-th SPSD kernel and $w^{(m)} \in \mathbb{R}_+$ is the associated linear combination weight. Following the term in [11], $\mu : \mathcal{L}^2 \to [0,1]$ is used to represent a symmetric label compatibility function, which has the properties that $\mu(l,l') = \mu(l,l'), \forall l,l' \in \mathcal{L}$ and $\mu(l,l) = 0, \forall l \in \mathcal{L}$. The label compatibility function penalizes similar pixels being assigned with different/incompatible labels. A simple label compatibility function would be given by Potts model, that is $\mu(l,l') = \delta(l \neq l')$. The form of pairwise potential in (3) is very general, and can be used to represent many potentials of practical interest.

Mean field approximation is used in [11] for solving problem (2), which is considered to be state-of-the-art. A filter-based method is used to accelerate the computation of message passing step (the product of kernel matrix and a column vector). In the following two sections, we will briefly revisit mean field approximation and the filter-based method, especially their respective limitations.

B. Mean Field Approximation

In mean field approximation, a variational distribution $Q(\mathbf{x})$ is introduced to approximate the Gibbs distribution $P(\mathbf{x})$, in which the marginals for each variable in CRF, $\{Q_i(\cdot)\}_{i\in\mathcal{N}}$, are supposed to be independent to each other such that $Q(\mathbf{x})$ can be completely factorized as

$$Q(\mathbf{x}) = \Pi_{i\in\mathcal{N}} Q_i(x_i). \tag{4}$$

Over the distribution $Q(\mathbf{x})$, mean field approximation minimizes the KullbackLeibler (KL)-divergence $\mathrm{D}(Q\|P)$:

$$\begin{aligned}
\mathrm{D}(Q\|P) &= \sum_{\mathbf{x}\in\mathcal{L}^N} Q(\mathbf{x}) \log \frac{Q(\mathbf{x})}{P(\mathbf{x})} && \text{(5a)}\\
&= \sum_{\mathbf{x}\in\mathcal{L}^N} Q(\mathbf{x}) \log Q(\mathbf{x}) - \sum_{\mathbf{x}\in\mathcal{L}^N} Q(\mathbf{x}) \log \frac{1}{Z} \exp(-\mathrm{E}(\mathbf{x})) && \text{(5b)}\\
&= \sum_{\mathbf{x}\in\mathcal{L}^N} Q(\mathbf{x}) \log Q(\mathbf{x}) + \sum_{\mathbf{x}\in\mathcal{L}^N} Q(\mathbf{x})\mathrm{E}(\mathbf{x}) + \log Z && \text{(5c)}
\end{aligned}$$

Recall the definition of the energy function for fully-connected pairwise CRFs in (2) and the complete factorization (4), we further have that

$$\begin{aligned}
\mathrm{D}(Q\|P) = &\sum_{i\in\mathcal{N}} \sum_{x_i\in\mathcal{L}} Q_i(x_i) \log Q_i(x_i) + \sum_{i\in\mathcal{N}} \sum_{x_i\in\mathcal{L}} Q_i(x_i)\psi_i(x_i)\\
&+ \sum_{i,j\in\mathcal{N}, i<j} \sum_{x_i,x_j\in\mathcal{L}} Q_i(x_i)Q_j(x_j)\psi_{i,j}(x_i, x_j) + \log Z,
\end{aligned} \tag{6}$$

To optimize over the i-th marginal, the KL-divergence $\mathrm{D}(Q\|P)$ is viewed as a function of $Q_i(\cdot)$ while keeping other marginals fixed:

$$\mathrm{D}(Q\|P) = \sum_{x_i\in\mathcal{L}} Q_i(x_i) \log Q_i(x_i) + \sum_{x_i\in\mathcal{L}} Q_i(x_i) \Bigg(\psi_i(x_i) + \sum_{j\in\mathcal{N}, j\neq i} \sum_{x_j\in\mathcal{L}} Q_j(x_j)\psi_{i,j}(x_i, x_j) \Bigg) + \text{const.} \tag{7}$$

It is easy to find out that minimizing $\mathrm{D}(Q\|P)$ w.r.t. $Q_i(\cdot)$ gives the following close-formed solutions, namely *mean field equations*:

$$Q_i(x_i) = \frac{1}{Z_i} \exp\Big(- \psi_i(x_i) + \sum_{j\in\mathcal{N}, j\neq i} \sum_{x_j\in\mathcal{L}} Q_j(x_j)\psi_{i,j}(x_i, x_j)\Big), \ \forall i \in \mathcal{N}, \tag{8}$$

where Z_i is the local normalization factor such that $\sum_{x_i\in\mathcal{L}} Q_i(x_i) = 1$. Updating the above mean field equations iteratively results in a monotonically decreased $\mathrm{D}(Q\|P)$.

One significant limitation of mean field approximation is that it may converge to one of potentially many local optima, because the variational problem to be optimized may be non-convex. A consequence of this non-convexity is that mean field is often sensitive to the initialization of Q.

C. Filter-based Matrix-vector Product

Recall the pairwise terms defined by SPSD kernels as in (3), the mean field equations (8) can be further expressed as:

$$\mathrm{Q}_i(l) = \frac{1}{Z_i} \exp\Big(- \psi_i(l) - \sum_{l'\in\mathcal{L}} \mu(l,l') \sum_{m=1}^{M} w^{(m)} \sum_{j\in\mathcal{N}, j\neq i} \mathrm{k}^{(m)}(\mathbf{f}_i, \mathbf{f}_j)\mathrm{Q}_i(l')\Big). \tag{9}$$

The computational bottleneck in updating the above equation can be expressed as the matrix-vector products $\mathbf{K}^{(m)}\mathbf{q}$, $m = 1, \cdots, M$, where $\mathbf{K}^{(m)} \in \mathcal{S}^N_+$ denotes the kernel matrix corresponding to $\mathrm{k}^{(m)}$, that is $K^{(m)}_{i,j} = \mathrm{k}^{(m)}(\mathbf{f}_i, \mathbf{f}_j)$, and $\mathbf{q} \in \mathbb{R}^N$ denotes a column vector made up by Q, that is $\mathbf{q} = [\mathrm{Q}_1(l), \cdots, \mathrm{Q}_N(l)]^\top$, $\forall l \in \mathcal{L}$. The naive implementation of the matrix-vector product needs $\mathcal{O}(N^2)$ time. Krähenbühl and Koltun [11] proposed to use a filter-based approach to compute the matrix-vector product in $\mathcal{O}(N)$ time, which will be discussed in the next section.

Filter-based methods [19] have been used in [11] to speed up the above matrix-vector product. The method in [11] is based on the assumption that pairwise potentials are Gaussian kernels:

$$\mathrm{k}^{(m)}(\mathbf{f}_i,\mathbf{f}_j) = \exp\left(-\frac{1}{2}(\mathbf{f}_i - \mathbf{f}_j)^\top \mathbf{\Lambda}^{(m)}(\mathbf{f}_i - \mathbf{f}_j)\right), \tag{10}$$

where $\mathbf{\Lambda}^{(m)} \in \mathcal{S}^D_+$, $m = 1, 2, \cdots, M$. The product of a Gaussian kernel matrix and an arbitrary column vector can be expressed as a Gaussian convolution w.r.t. $\mathbf{\Lambda}^{(m)}$ in feature space (see [19], [11] for more details). From the viewpoint of signal processing, the Gaussian convolution can be seen as a low-pass filter over the feature space. Then the convolution result can be recovered from a set of samples whose spacing is proportional to the standard deviation of the filter. A number of filtering methods [20], [19] can be used to compute the convolution efficiently, in which the computational complexity and memory requirement are both linear in N.

Filter-based approaches have a number of limitations, however:
(i) In general, the pairwise potentials are limited to Gaussian kernels over a Euclidean feature space.
(ii) The feature dimension cannot be very high. The bilateral filtering method in [20] has an exponential complexity in the dimension D. The time complexity of permutohedral lattice [19] is quadratic in D, which works well only when the input dimension is $5 \sim 20$. Beacause it does not create new lattice points during the blur step, accuracy penalty is accumulated with the growth of feature dimension.

D. Semidefinite Programming and SDCut Algorithms

Semidefinite programming (SDP) is a class of convex optimization problems, which minimize/maximize a linear objective function over the intersection of the cone of positive semidefinite matrices with an affine space. A general SDP problem can be expressed in the following form:

$$\min_{\mathbf{Y}\in\mathcal{S}^n_+} \quad \mathrm{p}(\mathbf{Y}) := \langle \mathbf{Y}, \mathbf{A}\rangle, \tag{11a}$$

$$\text{s.t.} \quad \langle \mathbf{Y}, \mathbf{B}_i\rangle = b_i, \ i = 1, 2, \cdots, q, \tag{11b}$$

where $\mathbf{A}, \{\mathbf{B}_i\}_{i=1,\cdots,q} \in \mathcal{S}^n$, $\mathbf{b} \in \mathbb{R}^n$, q and n are positive integers denoting the number of linear constraints and the dimension of matrix variables respectively.

SDP relaxation is widely incorporated to develop approximation algorithms for binary quadratic program (BQP), which optimizes a quadratic objective function over binary variables $\mathbf{y} \in \{0,1\}^n$. SDP-based approximation algorithms typically solve the BQP in the following two steps:
(i) Lift the binary variable $\mathbf{y}$ to a positive semidefinite matrix variable $\mathbf{Y} := \mathbf{y}\mathbf{y}^\top$ and solve the relaxed SDP problem over $\mathbf{Y}$ to a certain accuracy.
(ii) Round the SDP solution to obtain an approximated solution to the original BQP.

SDP problems can be solved by standard interior-point methods, which can be found in a number of optimization toolboxes, such as SeDuMi [21], SDPT3 [22] and MOSEK [23]. Although accurate and stable, interior-point methods scale poorly to the matrix dimension n and the number of linear constraints q. The computational complexity of interior-point methods for SDP problems is $\mathcal{O}(m^3 + mn^3 + m^2n^2)$ at each iteration in worst-case, and the associated memory requirement is $\mathcal{O}(m^2 + n^2)$.

The method proposed in [16], denoted as SDCut, can be used to solve the SDP problem (11) approximately yet efficiently. SDCut solves the following approximation of (11) using quasi-Newton

$$\min_{\mathbf{Y} \in \mathcal{S}^n_+} \quad \mathrm{p}_\gamma(\mathbf{Y}) := \langle \mathbf{Y}, \mathbf{A} \rangle + \frac{1}{2\gamma}(\|\mathbf{Y}\|_F^2 - \eta^2), \tag{12a}$$

$$\text{s.t.} \quad \langle \mathbf{Y}, \mathbf{B}_i \rangle = b_i, \; i = 1, 2, \cdots, q, \tag{12b}$$

where $\gamma > 0$ is a penalty parameter. Assuming that the constraints (12b) encode $\mathrm{trace}(\mathbf{Y}) = \eta$, where η is a constant defined by the problem itself, the above approximation have the following properties:

Proposition 1. *The following results holds: (i)* $\forall\, \epsilon > 0$, $\exists\, \gamma > 0$ *such that* $|\mathrm{p}(\mathbf{Y}^\star) - \mathrm{p}(\mathbf{Y}^\star_\gamma)| \leq \epsilon$*, where* $\mathbf{Y}^\star$ *denotes the optima for* (11) *and* $\mathbf{Y}^\star_\gamma$ *denotes that for* (12) *w.r.t.* γ*. (ii) For* $\gamma_2 > \gamma_1 > 0$*, we have* $\mathrm{p}(\mathbf{Y}^\star_{\gamma_1}) \geq \mathrm{p}(\mathbf{Y}^\star_{\gamma_2})$*, where* $\mathbf{Y}^\star_{\gamma_1}$ *and* $\mathbf{Y}^\star_{\gamma_2}$ *are the optimal solutions of* (12) *for* γ_1 *and* γ_2 *respectively.*

Proof. These results rely on the properties that $\mathrm{trace}(\mathbf{Y}) = \eta$. See [16] for details. □

The above results show that (12) is an accurate approximation to the problem (11), as the solution to (12) can be sufficiently close to that to (11) given a large enough γ. The advantage of (12) is that it has a much simpler Lagrangian dual:

Proposition 2. *The Lagrangian dual problem of* (12) *can be simplified to*

$$\max_{\mathbf{u} \in \mathbb{R}^q} \quad \mathrm{d}_\gamma(\mathbf{u}) := -\frac{\gamma}{2}\|(\mathbf{C}(\mathbf{u}))_+\|_F^2 - \mathbf{u}^\top \mathbf{b} - \frac{\eta^2}{2\gamma} \tag{13}$$

where $\mathbf{C}(\cdot) : \mathbb{R}^q \to \mathcal{S}^n$ *is defined as* $\mathbf{C}(\mathbf{u}) := -\mathbf{A} - \sum_{i=1}^q u_i \mathbf{B}_i$*, and* $(\cdot)_+ : \mathcal{S}^n \to \mathcal{S}^n_+$ *is defined as* $(\mathbf{Y})_+ = \mathbf{\Gamma}\mathrm{Diag}(\max(\mathbf{0}, \boldsymbol{\lambda}))\mathbf{\Gamma}^\top$*.* $\boldsymbol{\lambda} := [\lambda_1, \ldots, \lambda_n]^\top$ *and* $\mathbf{\Gamma}$ *stand for the respective eigenvalues and eigenvectors of* $\mathbf{Y}$*, that is* $\mathbf{Y} = \mathbf{\Gamma}\mathrm{Diag}(\boldsymbol{\lambda})\mathbf{\Gamma}^\top$*. The relationship between the optimal solution to the primal* (12) $\mathbf{Y}^\star$ *and the solution to the dual* (13) $\mathbf{u}^\star$ *is:* $\mathbf{Y}^\star = \gamma(\mathbf{C}(\mathbf{u}^\star))_+$*.*

Proof. See [16] for details. □

It is easy to find that the Lagrangian dual problem (13) is convex, and the p.s.d. matrix variable is eliminated in the dual. It is also proved in [16] that the simplified dual problem has the following nice properties:

Proposition 3. $\forall \mathbf{u} \in \mathbb{R}^q$, $\forall \gamma > 0$, $\mathrm{d}_\gamma(\mathbf{u})$ *yields a lower-bound on the optimal objective function value of the problem* (11).

Proposition 4. $\mathrm{d}(\cdot)$ *is continuously differentiable but not necessarily twice differentiable, and its gradient is given by*

$$\nabla \mathrm{d}_\gamma(\mathbf{u}) = -\gamma \left[\langle (\mathbf{C}(\mathbf{u}))_+, \mathbf{B}_1 \rangle, \cdots, \langle (\mathbf{C}(\mathbf{u}))_+, \mathbf{B}_q \rangle\right]^\top - \mathbf{b}. \tag{14}$$

Such that Wang *et al.* [16] adopted quasi-Newton methods to solve the dual problem (13). At each iteration of quasi-Newton methods, only the objective function d_γ and its gradient (14) need to be computed, where the computational bottleneck is on the calculation of $(\mathbf{C}(\mathbf{u}))_+$, which is equivalent to obtaining all the positive eigenvalues and the corresponding eigenvectors of $\mathbf{C}(\mathbf{u})$. Note that although the SDP problem discussed in this paper contains only linear equality constraints, the SDCut method and the proposed method can both easily extended to SDP problems with linear inequality constraints.

III. Matrix-vector Product based on Low-rank Approximation

One key contribution of this paper is the use of a low-rank approximation to the positive semidefinite kernel matrix, based on which low-rank quasi-Newton methods are developed for large-scale SDP CRF inference. We propose to approximate an SPSD kernel matrix $\mathbf{K} \in \mathcal{S}_+^N$ by a low-rank representation: $\mathbf{K} \approx \mathbf{\Phi}\mathbf{\Phi}^\top$, where $\mathbf{\Phi} \in \mathbb{R}^{N \times R_K}$ and $R_K \ll N$, such that both of the computational complexity and memory requirement for computing the aforementioned matrix-vector product are linear in N. Compared to [20], [19], the pairwise potential function is generalized to any positive semidefinite kernel function and there is no restriction on the input feature dimension.

The best quality can be achieved by a low-rank approximation depends on the spectral distribution of the kernel matrix itself, which is related to the smoothness (differentiability and Lipschitz continuity) of the underlying kernel function (see [24], [25], [26], [27], [28], [29] for more details). In general, eigenvalues of smooth kernels (*e.g.* Gaussian kernel) decay quickly and thus can be well approximated by low-rank matrices.

The optimal low-rank approximation in terms of both the spectral norm and Frobenius norm can be obatined by eigen-decomposition, while it is computationally inefficient whose computational complexity is generally cubic in N. There are a number of low-rank approximation methods achieving linear complexity in N, including Nyström methods [30], [31], [32], incomplete Cholesky decomposition [33], [34], random Fourier features [35], [36], and homogeneous kernel maps [37]. For detailed discussion, please refer to the review papers [38], [39], [40]. We adopt Nyström methods [37] in this paper for the low-rank approximation of kernel matrices.

Nyström methods can be used to approximate a positive semidefintie matrix $\mathbf{K} \in \mathcal{S}_+^N$, by sampling $R_0 \ll N$ columns of $\mathbf{K}$ (refer to as landmarks). Firstly $\mathbf{K}$ is expressed as:

$$\mathbf{K} = \begin{bmatrix} \mathbf{W} & {\mathbf{K}_{2,1}}^\top \\ \mathbf{K}_{2,1} & \mathbf{K}_{2,2} \end{bmatrix}, \tag{15}$$

where $\mathbf{W} \in \mathcal{S}^{R_0}$ denotes the intersection of the sampled R_0 columns and rows. The matrix $\mathbf{K}_{2,2} \in \mathcal{S}^{N-R_0}$ can be approximated as:

$$\mathbf{K}_{2,2} \approx \mathbf{K}_{2,1}\mathbf{\Gamma}_R\mathbf{\Sigma}_R^{-1}\mathbf{\Gamma}_R^\top{\mathbf{K}_{2,1}}^\top, \tag{16}$$

where $R \leq R_0$ and $\mathbf{\Sigma}_R = \mathrm{Diag}([\lambda_1, \ldots, \lambda_R]^\top)$. $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_R > 0$ are the R-largest eigenvalues of $\mathbf{W}$ and $\mathbf{\Gamma}_R$ contains the corresponding (column) eigenvectors. Note that $\mathbf{\Gamma}_R\mathbf{\Sigma}_R\mathbf{\Gamma}_R^\top$ is the best rank-R approximation to $\mathbf{W}$. Then we have a rank-R approximation to $\mathbf{K}$:

$$\mathbf{K} \approx \left(\begin{bmatrix} \mathbf{W} \\ \mathbf{K}_{2,1} \end{bmatrix}\mathbf{\Gamma}_R\mathbf{\Sigma}_R^{-\frac{1}{2}}\right)\left(\begin{bmatrix} \mathbf{W} \\ \mathbf{K}_{2,1} \end{bmatrix}\mathbf{\Gamma}_R\mathbf{\Sigma}_R^{-\frac{1}{2}}\right)^\top, \tag{17}$$

which is proved to have a bounded error to the optimal rank-R approximation given by the eigen-decomposition [39].

There are several strategies to sample representative landmarks, *i.e.*, columns of $\mathbf{K}$, including the standard uniform sampling [31], non-uniform sampling [32] and k-means clustering [41]. In this paper, we adopt the k-means method in [41] to select landmarks. At each round of k-means, only R columns of $\mathbf{K}$, rather than the entire matrix $\mathbf{K}$, is required to be instantiated.

Note that for Nyström methods, the positive semidefinite matrix $\mathbf{K}$ to be approximated can be any m-th kernel matrix $\mathbf{K}^{(m)}$ or the summation $\sum_{m=1}^{M} w^{(m)}\mathbf{K}^{(m)}$.

IV. SDP Relaxation to MAP Estimation Problems

In this section, we introduce SDP relaxation to the problem (2). Throughout the main body of this paper, the label compatibility function is assumed to be given by Potts model, that is $\mu(l,l') = \delta(l \neq l')$. The SDP relaxation corresponding to an arbitrary label compatibility function is discussed in Section VIII-A.

By defining $\mathbf{X} \in \{0,1\}^{N \times L}$, $\mathbf{H} \in \mathbb{R}^{N \times L}$ and $\mathbf{K} \in \mathcal{S}_+^N$ as $X_{i,l} = \delta(x_i = l)$, $H_{i,l} = \psi_i(l)$ and $K_{i,j} =$

$\sum_{m=1}^{M} w^{(m)}\mathrm{k}^{(m)}(\mathbf{f}_i, \mathbf{f}_j)$, the objective function of (2) can be re-written as:

$$\begin{aligned}
\mathrm{E}(\mathbf{x}) &= \sum_{i\in\mathcal{N}} \psi_i(x_i) + \sum_{i,j\in\mathcal{N}, i<j} \mu(x_i,x_j) \sum_{m=1}^{M} w^{(m)}\mathrm{k}^{(m)}(\mathbf{f}_i, \mathbf{f}_j), && \text{(18a)}\\
&= \sum_{i\in\mathcal{N}, l\in\mathcal{L}} \psi_i(l)\delta(x_i = l) + \sum_{i,j\in\mathcal{N}, i<j} (1 - \delta(x_i = x_j))K_{i,j}, && \text{(18b)}\\
&= \sum_{i\in\mathcal{N}, l\in\mathcal{L}} \psi_i(l)\delta(x_i = l) - \frac{1}{2}\sum_{i,j\in\mathcal{N}}\sum_{l,l'\in\mathcal{L}} \delta(x_i = l)\delta(x_j = l')K_{i,j} + \frac{1}{2}\sum_{i,j\in\mathcal{N}} K_{i,j}, && \text{(18c)}\\
&= \langle \mathbf{H}, \mathbf{X}\rangle - \frac{1}{2}\langle \mathbf{X}\mathbf{X}^\top, \mathbf{K}\rangle + \frac{1}{2}\mathbf{1}^\top\mathbf{K}\mathbf{1}. && \text{(18d)}
\end{aligned}$$

Such that the problem (2) can be expressed as the following binary quadratic problem (BQP):

$$\begin{aligned}
\min_{\mathbf{X}\in\{0,1\}^{N\times L}} \quad & \tilde{\mathrm{E}}(\mathbf{X}) := \langle \mathbf{H}, \mathbf{X}\rangle - \frac{1}{2}\langle \mathbf{X}\mathbf{X}^\top, \mathbf{K}\rangle && \text{(19a)}\\
\text{s.t.} \quad & \textstyle\sum_{l=1}^{L} X_{i,l} = 1, \ \forall i \in \mathcal{N}, && \text{(19b)}
\end{aligned}$$

Note that there is a one-to-one correspondence between the set of $\mathbf{x} \in \mathcal{L}^N$ and the set of $\mathbf{X} \in \{0,1\}^{N\times L}$ satisfying (19b), and $\mathrm{E}(\mathbf{x}) = \tilde{\mathrm{E}}(\mathbf{X}) + \frac{1}{2}\mathbf{1}^\top\mathbf{K}\mathbf{1}$ for equivalent $\mathbf{x}$ and $\mathbf{X}$.

By introducing $\mathbf{Y} := \begin{bmatrix} \mathbf{I}_L \\ \mathbf{X} \end{bmatrix} \begin{bmatrix} \mathbf{I}_L \\ \mathbf{X} \end{bmatrix}^\top$, the corresponding SDP relaxation to problem (19) can be expressed as:

$$\begin{aligned}
\min_{\mathbf{Y}\in\mathcal{S}_+^{N+L}} \quad & \langle \mathbf{Y}, \frac{1}{2}\begin{bmatrix} \mathbf{0} & \mathbf{H}^\top \\ \mathbf{H} & -\mathbf{K} \end{bmatrix}\rangle, && \text{(20a)}\\
\text{s.t.} \quad & Y_{l,l} = 1, \ l \in \mathcal{L}, && \text{(20b)}\\
& \frac{1}{2}(Y_{l,l'} + Y_{l',l}) = 0, \ l \le l', l,l' \in \mathcal{L}, && \text{(20c)}\\
& \frac{1}{2}\textstyle\sum_{l=1}^{L}(Y_{i+L,l} + Y_{l,i+L}) = 1, \ i \in \mathcal{N}, && \text{(20d)}\\
& Y_{i+L,i+L} = 1, \ i \in \mathcal{N}. && \text{(20e)}
\end{aligned}$$

Clearly we have $\mathrm{trace}(\mathbf{Y}) = N + L$ which is implicitly encoded by the linear constraints, and $\mathrm{rank}(\mathbf{Y}) = L$ which is non-convex and dropped by the SDP relaxation.

In the above formulation, the objective function and all the constraints (20b), (20c), (20d), (20e) are linear in $\mathbf{Y}$. Therefore the problem (20) can be re-written in the general form of (11), in which $n = \eta = N + L$, $q = 2N + L(L+1)/2$, $\mathbf{A} = \frac{1}{2}\begin{bmatrix} \mathbf{0} & \mathbf{H}^\top \\ \mathbf{H} & -\mathbf{K} \end{bmatrix}$, and thus solved using SDCut [16].

V. Low-rank Quasi-Newton Methods For SDP Inference

In this section, we follow the method in [16] (denoted as SDCut) which solves general BQPs. *Several major improvements are proposed to make SDCut scalable to the large-scale energy minimization problem* (19)*, which is another key contribution of this paper.*

Although it is shown in [16] that SDCut already runs much faster than standard interior-point methods, there are still several issues to be addressed for the problem to be solved in this work:
(i) It is shown in [16] that $\mathrm{rank}((\mathbf{C}(\mathbf{u}))_+)$ drops significantly in the first several iterations, and Lanczos methods [42] can be used to efficiently compute a few leading eigenpairs. However, because $(\mathbf{C}(\mathbf{u}))_+$ is not necessarily low-rank in the initial several iterations, much of time may be spent on the first several eigen-decompositions. In the CRFs considered in this paper, there are up to $681{,}600$ variables. Using the original SDCut method, the time spent on the first several iterations can be prohibitive.
(ii) In general, a BFGS-like method has a superlinear convergence speed under the condition that the objective function is twice continuously differentiable. However, the dual objective function (13) is not necessarily twice differentiable. So *the convergence speed of SDCut is unknown*. In practice, SDCut usually needs more than 100 iterations to converge.

In the next two sections, we introduce two improvements to the SDCut method, which address the above two problems and increase the scalability of SDCut significantly. The improved method is refer to as LR-SDCut and its procedure is summarized in Algrithm 1.

Algorithm 1 LR-SDCut algorithm for MAP estimation.

Input: $\mathbf{A}$, $\{\mathbf{B}_i\}_{i=1,2,\cdots,q}$, $\mathbf{b}$, γ, $K_{\max}$, $\tau > 0$, $r \ll N$.
Initialization: $\mathbf{u}^{(0)} = \mathbf{0}$, $\tilde{\mathrm{E}}^\star = +\inf$, $\mathbf{A} = \mathbf{A} - \nu \mathbf{I}_N$ where ν is the r-th smallest eigenvalue of $\mathbf{A}$.
for $k = 0, 1, 2, \ldots, K_{\max}$ **do**
Step1: $\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} - \rho \mathbf{H} \nabla \mathrm{d}_\gamma(\mathbf{u}^{(k)})$, where $\mathbf{H}$ is updated to approximate $(\nabla^2 \mathrm{d}_\gamma(\mathbf{u}^{(k)}))^{-1}$ and $0 < \rho \leq 1$ is the step size.
Step2: $\mathbf{X}^{(k+1)} = \mathrm{Round}(\gamma(\mathbf{C}(\mathbf{u}^{(k+1)}))_+)$.
Step3: If $\tilde{\mathrm{E}}(\mathbf{X}^{(k+1)}) < \tilde{\mathrm{E}}^\star$, $\mathbf{X}^\star = \mathbf{X}^{(k+1)}$.
Step4: Exit, if $\big(\mathrm{d}_\gamma(\mathbf{u}^{(k+1)}) - \mathrm{d}_\gamma(\mathbf{u}^{(k)})\big) / \max\{|\mathrm{d}_\gamma(\mathbf{u}^{(k+1)})|, |\mathrm{d}_\gamma(\mathbf{u}^{(k)})|, 1\} \leq \tau$.

Output: $\mathbf{X}^\star$, $\tilde{\mathrm{E}}^\star$.

A. A Low-rank Initial Point

If the initialization of the dual variable $\mathbf{u}^{(0)}$ is $\mathbf{0}$, then we have $\mathbf{C}(\mathbf{u}^{(0)}) = -\mathbf{A}$. Without affecting the optimal solution to (11), $\mathbf{A}$ can be perturbed so as to reduce $\mathrm{rank}((\mathbf{C}(\mathbf{u}^{(0)}))_+)$ to a small integer, based on:
(i) For $\mathbf{Y} \in \mathcal{S}^n_+ \cap \{\mathrm{trace}(\mathbf{Y}) = n\}$, $\langle \mathbf{Y}, \mathbf{A} + \nu \mathbf{I}_n \rangle = \langle \mathbf{Y}, \mathbf{A} \rangle + \nu n$. So the matrix $\mathbf{A}$ in the problem (11) can be equivalently replaced by $\mathbf{A} + \nu \mathbf{I}_n$, $\forall \nu \neq 0$.
(ii) Suppose that $\lambda \neq 0$ and $\mathbf{x} \in \mathbb{R}^n$ is an eigenpair of $\mathbf{A} \in \mathcal{S}^n$, *i.e.*, $\mathbf{A}\mathbf{x} = \lambda \mathbf{x}$, then $\mathbf{A} + \nu \mathbf{I}_n$ has an eigenpair: $\lambda + \nu$ and $\mathbf{x}$, $\forall \nu \neq 0$.

To decrease the rank of $\big(\mathbf{C}(\mathbf{u}^{(0)})\big)_+$ to $r \ll n$, we can equivalently replace $\mathbf{A}$ by $\mathbf{A} - \nu \mathbf{I}_n$, where ν is the r-th smallest eigenvalue of $\mathbf{A}$.

B. Rounding Schemes and Early Stop

Traditionally, a feasible solution $\mathbf{X}$ to the BQP problem (19) is obtained by rounding the optimal solution $\mathbf{Y}^\star$ to the corresponding SDP formulation (12). The rounding procedure will be carried out until the quasi-Newton algorithm converges. In contrast, we perform the rounding procedure on the non-optimal solution $\mathbf{Y}^{(k)} := \gamma\big(\mathbf{C}(\mathbf{u}^{(k)})\big)_+$ at each iteration k of the quasi-Newton algorithm (Step2 in Algorithm 1). In practice, we find that the dual objective value of (13), *i.e*. the lower-bound to the optimal value of $\tilde{\mathrm{E}}(\mathbf{X})$, increases dramatically in the first several iterations. Simultaneously, the value of $\tilde{\mathrm{E}}(\mathbf{X}^{(k)})$ also drops significantly for the first several ks. This observation inspires us to stop the quasi-Newton algorithm long before convergence, without affecting the final solution quality.

In this work, we adopt the random rounding scheme proposed in [43] to derive $\mathbf{X}$ from $\mathbf{Y}^{(k)} := \gamma(\mathbf{C}(\mathbf{u}^{(k)}))_+$. Note that because $\mathbf{Y}^{(k)}$ is positive semidefinite, it can be decomposed to $\mathbf{Y}^{(k)} = \mathbf{\Psi}\mathbf{\Psi}^\top$, where $\mathbf{\Psi} \in \mathbb{R}^{N \times R_Y}$ and $R_Y = \mathrm{rank}(\mathbf{Y}^{(k)})$. The rounding scheme can be expressed in the following two steps:
(i) Random Projection: $\hat{\mathbf{X}} = \mathbf{\Psi}\mathbf{P}$, where $\mathbf{P} \in \mathbb{R}^{R_Y \times L}$ and each entry $P_{i,j}$ is independently sampled from the standard Gaussian distribution with mean 0 and variance 1, *i.e.*, $P_{i,j} \sim N(0,1)$.
(ii) Discretization: Obtain $\mathbf{X} \in \{0,1\}^{N \times L}$ by discretizing the above $\hat{\mathbf{X}}$, that is, $X_{i,l} = \delta(\hat{X}_{i,l} > \hat{X}_{i,l'}, \forall l' \in \mathcal{L}, l' \neq l)$.

C. Computational Complexity and Memory Requirement

The computational bottleneck of LR-SDCut is the eigen-decomposition of $\mathbf{C}(\mathbf{u})$ at each iteration, which is performed by Lanczos methods [42] in this paper. Lanczos methods only require users to implement the matrix-vector product $\mathbf{C}(\mathbf{u})\mathbf{d} = -\mathbf{A}\mathbf{d} - (\sum_{i=1}^q u_i \mathbf{B}_i)\mathbf{d}$, where $\mathbf{d} \in \mathbb{R}^n$ denotes a so-called "Lanczos vector" produced by Lanczos algorithms iteratively. In this section, we will how to accelerate the computation of this matrix-vector product by utilizing the specific structures of $\mathbf{A}$ and $\{\mathbf{B}_i\}_{i=1,\cdots,q}$, and then give the computational cost and memory requirement of LR-SDCut.

For the problem (20), $\mathbf{A} = \frac{1}{2}\begin{bmatrix} \mathbf{0} & \mathbf{H}^\top \\ \mathbf{H} & -\mathbf{K} \end{bmatrix}$, and $\{\mathbf{B}_i\}_{i=1,2,\cdots,q}$ have specific structures such that

$$\sum_{i=1}^q u_i \mathbf{B}_i = \begin{bmatrix} \mathrm{Diag}(\mathbf{u}_1) + \frac{1}{2}\mathrm{LTri}(\mathbf{u}_2) & \frac{1}{2}\mathbf{u}_3^\top \otimes \mathbf{1} \\ \frac{1}{2}\mathbf{u}_3 \otimes \mathbf{1}^\top & \mathrm{Diag}(\mathbf{u}_4) \end{bmatrix}, \tag{21}$$

where $\mathbf{u}_1 \in \mathbb{R}^L$, $\mathbf{u}_2 \in \mathbb{R}^{L(L-1)/2}, \mathbf{u}_3, \mathbf{u}_4 \in \mathbb{R}^N$ denote the respective dual variables w.r.t. constraints (20b), (20c), (20d), (20e) and such that $\mathbf{u} = \big[\mathbf{u}_1^\top, \mathbf{u}_2^\top, \mathbf{u}_3^\top, \mathbf{u}_4^\top\big]^\top$. $\mathrm{LTri}(\mathbf{u})\colon \mathbb{R}^{L(L-1)/2} \to \mathcal{S}^L$ produces an $L \times L$ symmetric matrix

whose lower triangular part is made up of the elements of the input vector $\mathbf{u} \in \mathbb{R}^{L(L-1)/2}$, that is $\text{LTri}(\mathbf{u}) = \begin{cases} 0 & \text{if } i=j \\ u_{(L-1)!/j!+i-j} & \text{if } i>j \\ u_{(L-1)!/i!+j-i} & \text{if } i<j \end{cases}$. Then the matrix-vector product $\mathbf{C}(\mathbf{u})\mathbf{d}$ can be expressed as:

$$\mathbf{C}(\mathbf{u})\mathbf{d} = \underbrace{-\frac{1}{2}\begin{bmatrix} \mathbf{H}^\top \mathbf{d}_2 \\ \mathbf{H}\mathbf{d}_1 - \mathbf{K}\mathbf{d}_2 \end{bmatrix}}_{\mathbf{A}\mathbf{d}:\,\mathcal{O}(NL+NR_K)} - \underbrace{\begin{bmatrix} \mathbf{u}_1 \circ \mathbf{d}_1 + \frac{1}{2}\text{LTri}(\mathbf{u}_2)\mathbf{d}_1 + \frac{1}{2}(\mathbf{u}_3^\top \mathbf{d}_2)\mathbf{1} \\ \frac{1}{2}(\mathbf{1}^\top \mathbf{d}_1)\mathbf{u}_3 + \mathbf{u}_4 \circ \mathbf{d}_2 \end{bmatrix}}_{(\sum_{i=1}^q u_i \mathbf{B}_i)\mathbf{d}:\,\mathcal{O}(L^2+N)}, \tag{22}$$

where $\mathbf{d}_1 \in \mathbb{R}^L, \mathbf{d}_2 \in \mathbb{R}^N$ and such that $\mathbf{d} = \left[\mathbf{d}_1^\top, \mathbf{d}_2^\top\right]^\top$. Accordingly, the computational cost of solving (20) by LR-SDCut at each descent iteration, that is the complexity of eigen-decomposition of $\mathbf{C}(\mathbf{u})$, is:

$$\underbrace{\mathcal{O}\Big((N+L)R_Y^2 + \underbrace{(NR_K + NL + L^2)}_{\text{matrix-vector product (22)}} R_Y \Big)}_{\text{Lanczos factorization}} \times \#\text{Lanczos-Iters}, \tag{23}$$

and the memory requirement is $\mathcal{O}(N(L + R_Y + R_K) + LR_Y)$, where R_K and R_Y denotes the rank of $\mathbf{K}$ and $(\mathbf{C}(\mathbf{u}))_+$ respectively. Note that the computational complexity is linear in the number of CRF variables N, which is the same as mean field approximation.

VI. Applications

To show the superiority of the proposed method, we evaluate it and other methods on two applications in this section: image segmentation and image co-segmentation. In the following our experiments, the maximum number of iterations $K_{\max}$ for LR-SDCut is set to 10; the initial rank r is set to 20; and the penalty parameter γ is set to 1000.

A. Application 1: Image Segmentation

Following the work in [11], pairwise potentials for image segmentation are expressed in the following form:

$$K_{i,j}^{(1)} = \exp\left(-\frac{|\mathbf{p}_i - \mathbf{p}_j|^2}{2\theta_\alpha^2} - \frac{|\mathbf{c}_i - \mathbf{c}_j|^2}{2\theta_\beta^2} \right), \tag{24}$$

where $\mathbf{p}_i$ and $\mathbf{c}_i$ are the position and color value of pixel i respectively, and similarly for $\mathbf{p}_j$ and $\mathbf{c}_j$. The matrix defined in (24) corresponds to the appearance kernel which penalizes the case that two adjacent pixels with similar color and different labels. The label compatibility function is given by the Potts model $\mu(l,l') = \delta(l \neq l')$.

The kernel matrix $\mathbf{K}^{(1)}$ can be decomposed to the hadamard product of two independent kernel matrices: $\mathbf{K}^{(1)} = \mathbf{K}_p^{(1)} \circ \mathbf{K}_c^{(1)}$, where $k_p^{(1)}(\mathbf{f}_i,\mathbf{f}_j) = \exp\left(\frac{-|\mathbf{p}_i - \mathbf{p}_j|^2}{2\theta_\alpha^2}\right)$ and $k_c^{(1)}(\mathbf{f}_i,\mathbf{f}_j) = \exp\left(\frac{-|\mathbf{c}_i - \mathbf{c}_j|^2}{2\theta_\beta^2}\right)$.

Nyström methods are performed on $\mathbf{K}_p^{(1)}$ and $\mathbf{K}_c^{(1)}$ individually: $\mathbf{K}_p^{(1)} \approx \mathbf{\Phi}_p \mathbf{\Phi}_p^\top$ and $\mathbf{K}_c^{(1)} \approx \mathbf{\Phi}_c \mathbf{\Phi}_c^\top$, where $\mathbf{\Phi}_p \in \mathbb{R}^{N \times R_p}$ and $\mathbf{\Phi}_c \in \mathbb{R}^{N \times R_c}$. Then we have:

$$\mathbf{K}^{(1)}\mathbf{d} = (\mathbf{K}_p^{(1)} \circ \mathbf{K}_c^{(1)})\mathbf{d} \tag{25a}$$

$$= \text{diag}\left(\mathbf{\Phi}_p \mathbf{\Phi}_p^\top \text{Diag}(\mathbf{d}) \mathbf{\Phi}_c \mathbf{\Phi}_c^\top \right) \tag{25b}$$

$$= \left(\left(\mathbf{\Phi}_p \mathbf{\Phi}_p^\top \left(\text{Diag}(\mathbf{d}) \mathbf{\Phi}_c \right) \right) \circ \mathbf{\Phi}_c \right) \mathbf{1}. \tag{25c}$$

This computation requires $\mathcal{O}(NR_cR_p)$ operations (R_c and R_p are set to 20 and 10 respectively). Performing Nyström on $\mathbf{K}_p^{(1)}$ and $\mathbf{K}_c^{(1)}$ separately instead of on $\mathbf{K}^{(1)}$ directly brings two benifits: (i) the memory requirement is reduced from R_cR_p to $R_c + R_p$; (ii) For multiple images with the same resolution, we only need to perform Nyström on $\mathbf{K}_p^{(1)}$ once, as the input features (positions $\mathbf{p}_i, i = 1, \cdots, N$) are the same.

The improved Nyström method [41] is adopted to obtain the low rank approximation of $\mathbf{K}_c^{(1)}$ and $\mathbf{K}_p^{(1)}$. As in [41], K-means clustering is used to select representative landmarks.

Experiments The proposed algorithm is compared with mean field on MSRC 21-class database. The test data are 93 representative images with accurate ground truth provided by [11]. The unary potentials are also obtained

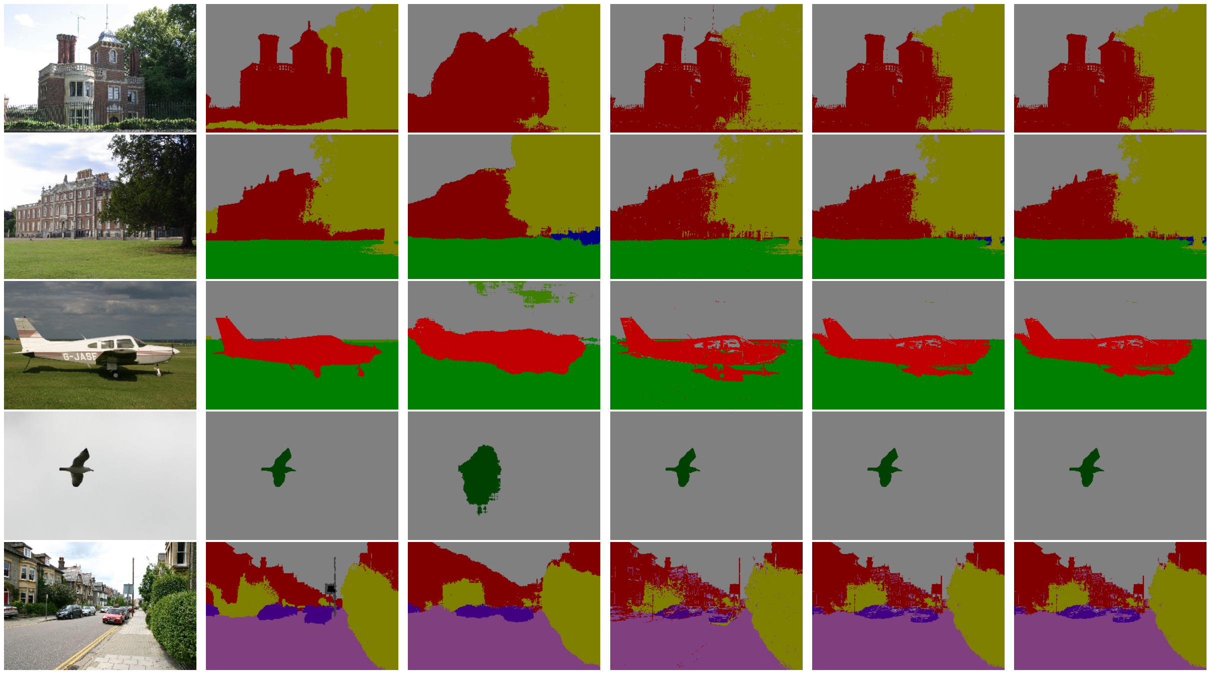

Fig. 1: Qualitative results of image segmentation. Original images and the corresponding ground truth are shown in the first two columns. The third column demonstrates the segmentation results based only on unary terms. The results of mean field methods with different matrix-vector product approaches are illustrated in the fourth and fifth columns. Our methods achieves similar visual performance with mean field methods.

	Unary	MF+filter	MF+Nys.	LR-SDCut
Time(s)	NA	0.29	6.6	74
Accu.	0.79	0.83	0.83	0.83
Energy	$1.29 \cdot 10^5$	$9.79 \cdot 10^4$	$1.15 \cdot 10^5$	$\mathbf{9.02 \cdot 10^4}$

TABLE II: Quantitative results of image segmentation. Our method runs slower than mean field methods but gives significantly lower energy. Unfortunately, the lower energy does not lead to better segmentation accuracy.

from [11]. The parameters θ_α, θ_β and $w^{(1)}$ are set to 60, 20 and 10 respectively. The iteration number limit for mean field inference is set to 20. All experiments are conducted using a single CPU with 10GB memory. As for the matrix-vector product in the mean field method, both the filter-based and Nyström-based approaches are evaluated (refer to as MF+filter and MF+Nys. respectively). The evaluated images have around 60,000 pixels and so the number of MRF variables is also around 60,000 for each image.

Fig. 1 shows the qualitative results for image segmentation. We can see that our method achieves similar results to the mean field approach. In Table II, quantitative results are demonstrated. Althgouh the computational complexity of mean field and our method are both linear in N, mean field is still faster than ours in this experiment. This is partially because the code of mean field is highly optimized using C++, while ours is unoptimized. A speed up is expected if our code is further optimized and parallelized. Note that the filter-based method [19] can be also incorporated into our algorithm to compute matrix-vector products, which is likely to be faster than Nyström methods but limited to Gaussian kernels in general.

Despite the slower speed, *our method achieves significantly lower energy than mean field*, which means our method is better from the viewpoint of MAP estimation. Unfortunately, the superiority of our method in terms of optimization does not lead to better segmentation performance. Actually, all of the evaluated methods have similar segmentation accuracy.

B. Application 2: Image Co-segmentation

The image co-segmentation problem requires that the same object be segmented from multiple images. There are two optimization criteria: the color and spatial consistency within one image and the separability between foreground

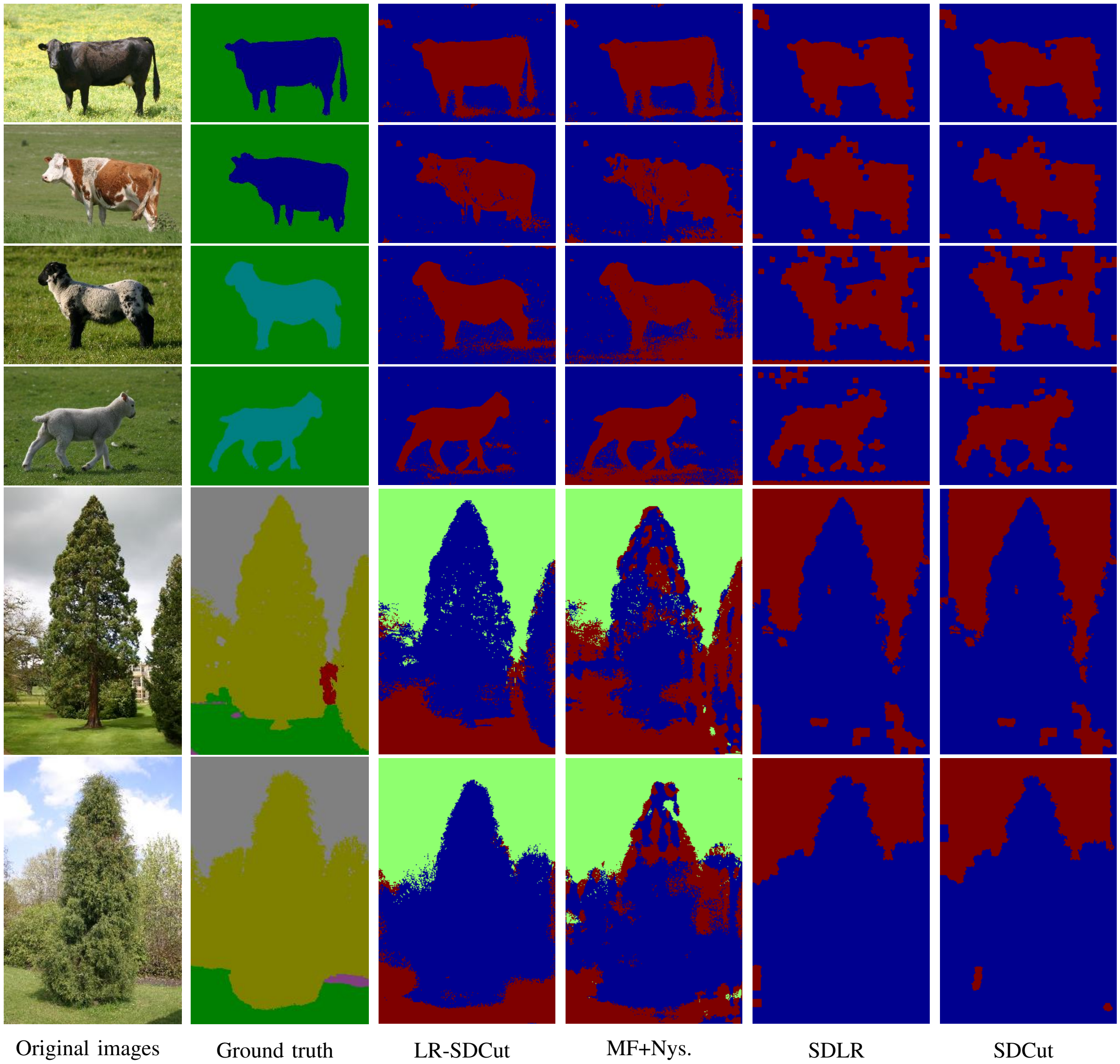

Fig. 2: Qualitative results for image co-segmentation. Three classes of objects from MSRC datasets are used for the evaluation. Our approach and Mean Field (MF+Nys.) are performed on the original pixel-level images. Because SDLR [44] and SDCut [44] cannot scale up to pixel-level images, they are evaluated on superpixels. Our method performs best visually. We randomly repeat mean field approximation 5 times for each dataset and select the best result. Mean field is not stable at this task and sometimes converges to an undesirable local optimal point (see "tree" for example). SDLR and SDCut achieve worse results than our's, since some image details are lost due to the use of superpixels.

Data	#pics	N	LR-SDCut	MF+Nys.	N	SDLR	SDCut
Cow	10	681600	**1415**	1965	6713	9530	307
Sheep	8	545280	**1066**	2045	5375	6932	583
Tree	9	613440	**1137**	1490	6026	1090	1316

TABLE III: Running times for image co-segmentation. Our method is slightly faster than mean field. The number of MRF variables N for two groups of evaluated methods are shown in the third and sixth columns. The problems solved by our approach are much larger than those of SDLR and SDCut.

and background over all images. There is no unary potentials for image co-segmentation and the pairwise potentials

	LR-SDCut	MF+Nys.	SDLR	SDCut
Cow	$\mathbf{0.73}(\mathbf{-1.59 \cdot 10^5})$	$0.67(-1.58 \cdot 10^5)$	0.66	0.69
Sheep	$\mathbf{0.74}(\mathbf{-8.07 \cdot 10^4})$	$0.49(-6.87 \cdot 10^4)$	0.57	0.58
Tree	$\mathbf{0.83}(\mathbf{-2.23 \cdot 10^5})$	$0.65(-2.03 \cdot 10^5)$	0.66	0.68

TABLE IV: Segmentation accuracy (energy) of image co-segmentation. Our method and Mean field work on original pixels, while SDLR and SDCut work on superpixels. For all the three evaluated datasets, our method achieves the lowest energies and highest segmentation scores.

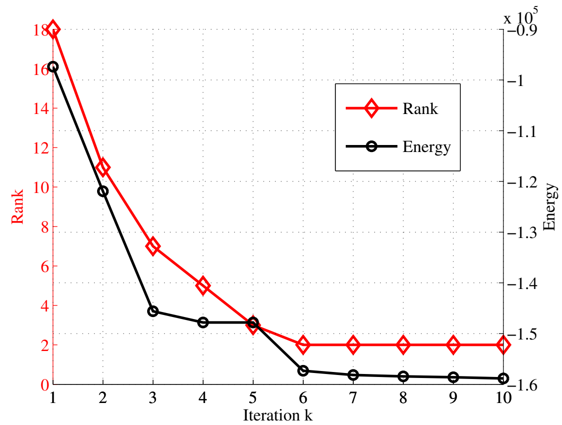

Fig. 3: Rank and Energy at each iteration for co-segmentation on the "cow" data set. Both of the rank of $(\mathbf{C}(\mathbf{u}_k))_+$ and the energy of binary solution $\mathbf{y}_k$ decrease significantly in the first several ks.

are shown in the following:

$$K^{(1)}_{i,j} = \varphi_{i,j} \exp\left(-\frac{|\mathbf{p}_i - \mathbf{p}_j|^2}{2\theta_\alpha^2} - \frac{|\mathbf{c}_i - \mathbf{c}_j|^2}{2\theta_\beta^2}\right), \tag{26a}$$

$$\mathbf{K}^{(2)} = \mathbf{\Omega}_N(\kappa N \mathbf{I}_N + \tilde{\mathbf{K}}^{(2)})^{-1}\mathbf{\Omega}_N, \tag{26b}$$

where $\varphi_{i,j} = 1$ if pixels i and j locate in the same image; $\varphi_{i,j} = 0$, otherwise. $\kappa > 0$ is a regularization parameter. $\mathbf{K}^{(1)}$ is a block-diagonal matrix, and the matrix-vector product for $\mathbf{K}^{(1)}$ can be computed using the method described in Section VI-A. $\mathbf{K}^{(2)}$ is the inter-image discriminative clustering cost matrix (see [44] for details). $\mathbf{\Omega}_N = \mathbf{I}_N - \frac{1}{N}\mathbf{1}\mathbf{1}^\top$ is the centering projection matrix, and $\tilde{\mathbf{K}}^{(2)}$ is the χ^2 kernel matrix of sift features. $\tilde{\mathbf{K}}^{(2)}$ can be approximated by a low-rank decomposition: $\tilde{\mathbf{K}}^{(2)} \approx \tilde{\mathbf{\Phi}}\tilde{\mathbf{\Phi}}^\top$, where $\tilde{\mathbf{\Phi}} \in \mathbb{R}^{N \times R_{K_2}}$. Based on the matrix inversion lemma, we have:

$$\begin{aligned}\mathbf{K}^{(2)} &= \frac{1}{\kappa N}\mathbf{\Omega}_N\Big(\mathbf{I}_N - \underbrace{\tilde{\mathbf{\Phi}}(\kappa N \mathbf{I}_D + \tilde{\mathbf{\Phi}}^\top\tilde{\mathbf{\Phi}})^{-1}\tilde{\mathbf{\Phi}}^\top}_{\text{decompose to } \mathbf{\Phi}\mathbf{\Phi}^\top \text{ and } \mathbf{\Phi}\mathbf{\Phi}^\top\mathbf{1}=0}\Big)\mathbf{\Omega}_N \\ &= \frac{1}{\kappa N}(\mathbf{\Omega}_N - \mathbf{\Phi}\mathbf{\Phi}^\top).\end{aligned} \tag{27}$$

Through the above equation, the matrix-vector product for $\mathbf{K}^{(2)}$ can be computed efficiently in $\mathcal{O}(NR_{K_2})$ time (R_{K_2} is set to 640 in the experiments). The pairwise potentials are not necessarily submodular, because entries of $\mathbf{K}^{(2)}$ may be negative. Note that *the matrix-vector product for* $\mathbf{K}^{(2)}$ *cannot be performed by the filter-based method of [11], because* $\mathbf{K}^{(2)}$ *may not be a Gaussian kernel*.

Experiments Three groups of images are selected from the MSRC dataset for image co-segmentation. Besides our approach and mean field, the SDP-based algorithms in [44] (denoted as SDLR) and [16] (denoted as SDCut) are also evaluated. Our method and mean field are evaluated at the original pixel level, while SDLR and SDCut are evaluated only on superpixels.

The code for SDLR and SDCut is provided by authors of the original papers, where the default settings are used. The iteration limit for mean field is set to 100. To prevent mean field from converging to undesirable local optima, we randomly run the method 5 times. All experiments are conducted on a single CPU with 20GB memory. The *intersection-over-union* accuracy is used to measure the segmentation performance.

From the results illustrated in Fig. 2, we see that our approach achieves much more accurate co-segmentation results than both SDLR and SDCut. The performance of mean field is also worse than ours.

Table III demonstrates the number of variables and computational time for each method. The number of variables for the problem solved by our method and mean field is around 100 times larger than those for SDLR and SDCut. Our approach is slightly faster than mean field, and significantly more scalable than SDLR and SDCut.

The quantitative performance is shown in Table IV. Our approach achieves significantly better co-segmentation accuracy than all the other methods. As for energy, our approach also produces lower energies than mean field. Empirically, we found mean field is sensitive to initialization. Take "tree" as example, the difference is $5.3 \cdot 10^4$ between the best and worst energy in the 5 repeats of mean field with random initializations. If we repeat mean field 100 times, the best energy improves from $-2.03 \cdot 10^5$ to $-2.08 \cdot 10^5$, but still worse than ours ($-2.23 \cdot 10^5$).

Fig. 3 shows the change of $\mathrm{rank}((\mathbf{C}(\mathbf{u}^{(k)}))_+)$ and $\tilde{\mathrm{E}}(\mathbf{y}^{(k)})$ w.r.t. iteration k. Both of the rank and energy drops quickly in the first several iterations. Simultaneously, the lower-bound of the optimal energy $\tilde{\mathrm{E}}(\mathbf{y})$ (*i.e.* the dual objective value) increases from $-8.09 \cdot 10^7$ to $-4.36 \cdot 10^5$.

VII. Conclusion

In this paper, we have proposed an efficient, general method for the MAP estimation of fully-connected CRFs. The proposed SDP approach is more stable and accurate than mean field approximation, which is also more scalable than previous SDP methods. The use of low-rank approximation of the kernel matrix to perform matrix-vector products makes our approach even more efficient and applicable for any symmetric positive semidefinite kernel. In contrast, previous filter-based methods assume pairwise potentials to be based on a Gaussian or generalized RBF kernel. The computational complexity of our approach is linear in the number of CRF variables. The experiments on image co-segmentation validate that our approach can be applied on more general problems than previous methods.

As for future works, the proposed method can be parallelized to achieve even faster speed. The core of our method is quasi-Newton (or gradient descent) and eigen-decomposition, both of which can be parallelized on GPUs. Matrix-vector products, the main computational cost, can be implemented using CUDA function "cublasSgemm".

VIII. Appendix

A. SDP formulation for an arbitrary label compatibility function

For an arbitrary label compatibility function $\mu : \mathcal{L}^2 \to [0,1]$, with the properties that $\mu(l,l') = \mu(l',l), \forall l,l' \in \mathcal{L}$ and $\mu(l,l) = 0, \forall l \in \mathcal{L}$, the objective function of (2), $\mathrm{E}(\mathbf{x})$, can be re-written as follows:

$$
\begin{aligned}
\mathrm{E}(\mathbf{x}) &= \sum_{i \in \mathcal{N}} \psi_i(x_i) + \sum_{i,j \in \mathcal{N}, i<j} \mu(x_i,x_j) K_{i,j}, && \text{(28a)}\\
&= \sum_{i \in \mathcal{N}, l \in \mathcal{L}} \psi_i(l)\delta(x_i = l) + \sum_{i,j \in \mathcal{N}, i<j} \sum_{l,l' \in \mathcal{L}} \mu(l,l')\delta(x_i = l)\delta(x_j = l') K_{i,j}, && \text{(28b)}\\
&= \mathbf{h}^\top \mathbf{y} + \frac{1}{2}\mathbf{y}^\top \Big((\mathbf{U} + \mathbf{1}\mathbf{1}^\top) \otimes \mathbf{K} \Big) \mathbf{y}, && \text{(28c)}\\
&= \mathbf{h}^\top \mathbf{y} + \frac{1}{2}\mathbf{y}^\top (\mathbf{U} \otimes \mathbf{K}) \mathbf{y} + \frac{1}{2}\mathbf{1}^\top \mathbf{K}\mathbf{1}, && \text{(28d)}
\end{aligned}
$$

where $\mathbf{y} \in \{0,1\}^{NL}$, $\mathbf{h} \in \mathbb{R}^{NL}$, $\mathbf{U} \in \mathcal{S}^L$ are defined as $y_{(i-1)L-l} = \delta(x_i = l)$, $h_{(i-1)L-l} = \psi_i(l)$, $\forall i \in \mathcal{N}, l \in \mathcal{L}$ and $U_{l,l'} = \mu(l,l') - 1$, $\forall l, l' \in \mathcal{N}$. Such that the energy minimization problem (2) can be equivalently reformulated to the following binary quadratic problem:

$$
\begin{aligned}
\min_{\mathbf{y} \in \{0,1\}^{NL}} \quad & \hat{\mathrm{E}}(\mathbf{y}) := \mathbf{h}^\top \mathbf{y} + \frac{1}{2}\mathbf{y}^\top (\mathbf{U} \otimes \mathbf{K}) \mathbf{y}, && \text{(29a)}\\
\text{s.t.} \quad & \textstyle\sum_{l=1}^{L} y_{(i-1)L+l} = 1, \ \forall i \in \mathcal{N}, && \text{(29b)}
\end{aligned}
$$

Note that there is also a one-to-one correspondence between the set of $\mathbf{x} \in \mathcal{L}^N$ and the set of $\mathbf{y} \in \{0,1\}^{NL}$ satisfying (29b), and $\mathrm{E}(\mathbf{x}) = \hat{\mathrm{E}}(\mathbf{y}) + \frac{1}{2}\mathbf{1}^\top \mathbf{K}\mathbf{1}$ for equivalent $\mathbf{x}$ and $\mathbf{y}$.

By defining $\mathbf{Y} := \mathbf{y}\mathbf{y}^\top$, the SDP relaxation to (29) can be expressed as:

$$\min_{\mathbf{Y}\in\mathcal{S}_+^{NL}} \quad \langle \mathbf{Y}, \mathrm{Diag}(\mathbf{h}) + \frac{1}{2}\mathbf{U}\otimes\mathbf{K}\rangle, \tag{30a}$$

$$\text{s.t.} \quad \textstyle\sum_{l=1}^{L}(Y_{\substack{(i-1)L+l,\\(i-1)L+l}}) = 1, \, \forall i \in \mathcal{N}, \tag{30b}$$

$$\frac{1}{2}(Y_{\substack{(i-1)L+l,\\(i-1)L+l'}} + Y_{\substack{(i-1)L+l',\\(i-1)L+l}}) = 0, \;\; \forall l \neq l', l,l' \in \mathcal{L}, i \in \mathcal{N}, \tag{30c}$$

and we have $\mathrm{trace}(\mathbf{Y}) = N$ due to constraints (30b). The non-convex constraint $\mathrm{rank}(\mathbf{Y}) = 1$ is dropped by the above SDP relaxation. There are 1 constraint (30b) and $L(L-1)/2$ constraints (30c) for each $i \in \mathcal{N}$. The problem (30) can also be expressed in the form of (11), and solved by SDCut algorithm. In this case, $n = NL$, $\eta = N$, $q = N + NL(L-1)/2$, $\mathbf{A} = \frac{1}{2}\mathbf{U}\otimes\mathbf{K}$, and

$$\sum_{i=1}^{q} u_i \mathbf{B}_i = \mathrm{Diag}(\mathbf{u}_1)\otimes\mathbf{I}_L + \frac{1}{2}\begin{bmatrix} \mathrm{LTri}(\mathbf{u}_{2,1}) & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathrm{LTri}(\mathbf{u}_{2,N}) \end{bmatrix}, \tag{31}$$

where $\mathbf{u}_1 \in \mathbb{R}^N$ denotes the dual variables w.r.t. the constraints (30b), and $\mathbf{u}_{2,i} \in \mathbb{R}^{L(L-1)/2}$ corresponds to the constraints (30c) for each $i \in \mathcal{N}$. We also have $\mathbf{u} = [\mathbf{u}_1^\top, \mathbf{u}_{2,1}^\top, \cdots, \mathbf{u}_{2,N}^\top]^\top$.

Then the matrix-vector product $\mathbf{C}(\mathbf{u})\mathbf{d}$, $\forall \mathbf{d} \in \mathbb{R}^{NL}$, can be computed as:

$$\begin{aligned}\mathbf{C}(\mathbf{u})\mathbf{d} = &-\Big(\underbrace{\mathbf{h}\circ\mathbf{d} + \frac{1}{2}\mathcal{T}\Big(\mathbf{K}[\mathbf{d}_1,\cdots,\mathbf{d}_N]^\top\mathbf{U}\Big)}_{\mathbf{A}\mathbf{d}:\,\mathcal{O}(NLR_K+NL^2)}\Big) \\ &-\Big(\underbrace{(\mathbf{u}_1\otimes\mathbf{1})\circ\mathbf{d} + \frac{1}{2}\Big[\mathbf{d}_1^\top\mathrm{LTri}(\mathbf{u}_{2,1}),\ldots,\mathbf{d}_N^\top\mathrm{LTri}(\mathbf{u}_{2,N})\Big]^\top}_{(\sum_{i=1}^{q}u_i\mathbf{B}_i)\mathbf{d}:\,\mathcal{O}(NL^2)}\Big),\end{aligned} \tag{32}$$

where $\mathbf{d}$ is decomposed as $\mathbf{d} := [\mathbf{d}_1^\top, \mathbf{d}_2^\top, \ldots, \mathbf{d}_N^\top]^\top$, $\mathbf{d}_1, \cdots, \mathbf{d}_N \in \mathbb{R}^L$, and $\mathcal{T} : \mathbb{R}^{N\times L} \to \mathbb{R}^{NL}$ is defined as $\mathcal{T}(\mathbf{X}) = [X_{1,1}, \cdots, X_{1,L}, X_{2,1}, \cdots, X_{2,L}, \cdots, X_{N,L}]^\top$. Given that $\mathbf{K}$ has an R_K-rank approximation and $(\mathbf{C}(\mathbf{u}))_+$ has the rank of R_Y, the overall computational complexity of solving (30) using SDCut at each iteration is

$$\underbrace{\mathcal{O}\Big(NLR_Y^2 + \underbrace{\underbrace{(NLR_K + NL^2)}\quad R_Y}_{\text{matrix-vector product (32)}}\Big)}_{\text{Lanczos factorization}} \times \text{\#Lanczos-Iters}. \tag{33}$$

The corresponding memory requirement is $\mathcal{O}(NLR_Y + L^2 + NR_K)$. Note that the above formulation still need to be further validated by experiments.

References

[1] J. H. Kappes, B. Andres, F. A. Hamprecht, C. Schnorr, S. Nowozin, D. Batra, S. Kim, B. X. Kausler, J. Lellmann, N. Komodakis *et al.*, "A comparative study of modern inference techniques for discrete energy minimization problems," in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2013.

[2] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother, "A comparative study of energy minimization methods for Markov random fields with smoothness-based priors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 6, pp. 1068–1080, 2008.

[3] M. P. Kumar, V. Kolmogorov, and P. H. Torr, "An analysis of convex relaxations for MAP estimation of discrete MRFs," *J. Mach. Learn. Res.*, vol. 10, pp. 71–106, 2009.

[4] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 10, pp. 1568–1583, 2006.

[5] J. H. Kappes, B. Savchynskyy, and C. Schnorr, "A bundle approach to efficient map-inference by lagrangian relaxation," in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2012.

[6] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer, "Optimizing binary MRFs via extended roof duality," in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2007.

[7] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, 2001.

[8] V. Kolmogorov and R. Zabin, "What energy functions can be minimized via graph cuts?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 147–159, 2004.

[9] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient belief propagation for early vision,” *Int. J. Comp. Vis.*, vol. 70, no. 1, pp. 41–54, 2006.

[10] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky, “MAP estimation via agreement on trees: message-passing and linear programming,” *Information Theory, IEEE Transactions on*, vol. 51, no. 11, pp. 3697–3717, 2005.

[11] P. Krähenbühl and V. Koltun, “Efficient inference in fully connected CRFs with gaussian edge potentials,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2011.

[12] ——, “Parameter learning and convergent inference for dense random fields,” in *Proc. Int. Conf. Mach. Learn.*, 2013.

[13] Y. Zhang and T. Chen, “Efficient inference for fully-connected CRFs with stationarity,” in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2012.

[14] N. D. Campbell, K. Subr, and J. Kautz, “Fully-connected CRFs with non-parametric pairwise potentials,” in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2013.

[15] Q. Huang, Y. Chen, and L. Guibas, “Scalable semidefinite relaxation for maximum a posterior estimation,” in *Proc. Int. Conf. Mach. Learn.*, 2014.

[16] P. Wang, C. Shen, and A. Hengel, “A fast semidefinite approach to solving binary quadratic problems,” in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2013.

[17] S. I. Wang, R. Frostig, P. Liang, and C. D. Manning, “Relaxations for inference in restricted boltzmann machines,” in *International Conference on Learning Representations*, 2014.

[18] R. Frostig, S. I. Wang, P. S. Liang, and C. D. Manning, “Simple map inference via low-rank relaxations,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2014.

[19] A. Adams, J. Baek, and M. A. Davis, “Fast high-dimensional filtering using the permutohedral lattice,” in *EUROGRAPHICS*, 2010.

[20] S. Paris and F. Durand, “A fast approximation of the bilateral filter using a signal processing approach,” in *Proc. Eur. Conf. Comp. Vis.*, 2006.

[21] J. F. Sturm, “Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones,” *Optimizat. Methods & Softw.*, vol. 11, pp. 625–653, 1999.

[22] K. C. Toh, M. Todd, and R. H. Ttnc, “Sdpt3 – a matlab software package for semidefinite programming,” *Opti. Meth. Soft.*, vol. 11, pp. 545–581, 1999.

[23] *The MOSEK optimization toolbox for MATLAB manual. Version 7.0 (Revision 139)*, MOSEK ApS, Denmark.

[24] J. Reade, “Eigenvalues of positive definite kernels,” *SIAM Journal on Mathematical Analysis*, vol. 14, no. 1, pp. 152–157, 1983.

[25] J. READE, “Eigenvalues of smooth positive definite kernels,” *Proceedings of the Edinburgh Mathematical Society*, vol. 35, pp. 41–45, 1992.

[26] C.-H. Chang and C.-W. Ha, “On eigenvalues of differentiable positive definite kernels,” *Integral Equations and Operator Theory*, vol. 33, no. 1, pp. 1–7, 1999.

[27] J. Buescu and A. Paixao, “Eigenvalue distribution of mercer-like kernels,” *Mathematische Nachrichten*, vol. 280, no. 9-10, pp. 984–995, 2007.

[28] ——, “Eigenvalue distribution of positive definite kernels on unbounded domains,” *Integral Equations and Operator Theory*, vol. 57, no. 1, pp. 19–41, 2007.

[29] A. J. Wathen and S. Zhu, “On the spectral distribution of kernel matrices related to radial basis functions,” 2013.

[30] C. Williams and M. Seeger, “The effect of the input density distribution on kernel-based classifiers,” in *Proc. Int. Conf. Mach. Learn.*, 2000.

[31] ——, “Using the Nyström method to speed up kernel machines,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2001.

[32] P. Drineas and M. W. Mahoney, “On the Nyström method for approximating a gram matrix for improved kernel-based learning,” *J. Mach. Learn. Res.*, vol. 6, pp. 2153–2175, 2005.

[33] S. Fine and K. Scheinberg, “Efficient SVM training using low-rank kernel representations,” *J. Mach. Learn. Res.*, vol. 2, pp. 243–264, 2002.

[34] F. R. Bach and M. I. Jordan, “Kernel independent component analysis,” *J. Mach. Learn. Res.*, vol. 3, pp. 1–48, 2003.

[35] A. Rahimi and B. Recht, “Random features for large-scale kernel machines,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2007.

[36] ——, “Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2008.

[37] A. Vedaldi and A. Zisserman, “Efficient additive kernels via explicit feature maps,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 480–492, 2012.

[38] F. Bach, “Sharp analysis of low-rank kernel matrix approximations,” *J. Mach. Learn. Res.*, vol. 30, pp. 185–209, 2013.

[39] A. Gittens and M. W. Mahoney, “Revisiting the Nyström method for improved large-scale machine learning,” in *Proc. Int. Conf. Mach. Learn.*, 2013.

[40] T. Yang, Y.-F. Li, M. Mahdavi, R. Jin, and Z.-H. Zhou, “Nyström method vs random Fourier features: A theoretical and empirical comparison,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2012.

[41] K. Zhang, I. W. Tsang, and J. T. Kwok, “Improved Nyström low-rank approximation and error analysis,” in *Proc. Int. Conf. Mach. Learn.*, 2008.

[42] D. C. Sorensen, *Implicitly restarted Arnoldi/Lanczos methods for large scale eigenvalue calculations*. Springer, 1997.

[43] J. Briët, F. M. de Oliveira Filho, and F. Vallentin, “The positive semidefinite grothendieck problem with rank constraint,” in *Automata, Languages and Programming*. Springer, 2010, pp. 31–42.

[44] A. Joulin, F. Bach, and J. Ponce, “Discriminative clustering for image co-segmentation,” in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2010.